IN THE CLAIMS

1. (Currently amended) A method ~~for allocating core processor bandwidth, in a computer between routine processing and interrupt servicing, wherein the core processor includes an instruction queue mechanism staging instructions for processing, the method~~ comprising:

detecting an interrupt service request;

inserting interrupt servicing instructions into an instruction queue in response to detecting ~~responsive to~~ the interrupt service request ~~into the instruction queue mechanism~~ where inserting the interrupt servicing instructions into the instruction queue results in processor bandwidth being allocated between the inserted interrupt servicing instructions and other program instructions via concurrent in-line stating of the interrupt servicing instructions and other program instructions; and

~~processing~~ executing the interrupt servicing instructions and other program instructions ~~within the instruction queue mechanism including the inserted interrupt servicing instructions;~~.

2. (Currently amended) The method of claim 1 ~~in which~~ where the instruction queue ~~mechanism~~ includes an instruction cache and an instruction fetch unit ~~for fetching~~ to fetch instructions from the instruction cache, ~~said~~ the processing being performed in such manner as to decode the instructions into micro-opcodes and execute the micro-opcodes in one or more out-of-order execution units.

3. (Currently amended) The method of claim 2 ~~which further comprises~~ comprising recycling the executed micro-opcodes for optional re-execution thereof in the one or more out-of-order execution units by retiring the executed micro-opcodes including those micro-opcodes representing the inserted interrupt servicing instructions to the instruction cache.

4. (Currently amended) The method of claim 3 ~~wherein~~ where the executed micro-opcodes are retired to the instruction cache in order.

5. (Currently amended) The method of claim 1 ~~wherein said~~ where detecting ~~is of~~ comprises detecting plural interrupts by~~, which further comprises~~ prioritizing ~~such~~ the plural interrupts and inserting one or more instances of ~~such~~ the interrupt servicing instructions into the instruction queue ~~mechanism~~ in accordance with one or more predefined interrupt servicing allocation criteria.

6. (Currently amended) The method of claim 5 ~~wherein~~ where the one or more allocation criteria include the priority of the interrupts and the capacity of the processor to allocate bandwidth to interrupt servicing.

7. (Currently amended) The method of claim 6 ~~wherein said~~ where the prioritizing is dynamically responsive to changing allocation criteria.

8. (Currently amended) The method of claim 1 ~~wherein core~~ where the processor bandwidth is allocated to interrupt servicing without flushing the instruction queue ~~mechanism~~.

9. (Currently amended) The method of claim 1 ~~in which said~~ where the detecting is performed by an interrupt processor, which after ~~said~~ the detecting ~~further comprises~~ the method comprising:

at the interrupt processor, determining whether the detected interrupt service request is of a priority meeting one or more defined high-priority criteria and if so then signaling the ~~core~~ processor to perform ~~said~~ the inserting; and

at the core processor, responding to said signaling by performing said inserting and said processing.

10. (Currently amended) The method of claim 1 ~~in which said~~ where the detecting is performed by an interrupt processor, ~~which at the core processor further comprises~~ the method comprising:

determining that a natural ~~core~~ processor context switch is imminent, said determining including analyzing one or more instructions within the instruction queue for context switch-inducing instructions;

signaling the interrupt processor to make ready the highest priority interrupt service request; and

signaling the instruction queue mechanism to fetch the readied interrupt service request in advance of ~~the~~ a context switch responsive to the determining.

11. (Currently amended) A ~~digital~~ processor, ~~for use in a computer supporting one or more hardware interrupt inputs, the processor~~ comprising:

an instruction cache;

a ~~fetch-and-decode~~ fetch unit, ~~said fetch-and-decode unit fetching~~ to fetch instructions from ~~said~~ the instruction cache; ~~and decoding~~

a decode unit to decode the instructions into micro-opcodes;

a ~~dispatch-and-execute~~ dispatch unit ~~having one or more execution ports,~~ to schedule ~~said dispatch-and-execute unit scheduling and executing~~ the micro-opcodes for execution;

an execute unit ~~in said~~ to execute the micro-opcodes ~~one or more execution ports and thereafter retiring the micro-opcodes back into the instruction cache~~; and

an interrupt-handl~~inger mechanism responsive to one or more hardware interrupt inputs, said interrupt-handling mechanism instructing~~ to signal ~~said~~ the ~~fetch-and-decode~~ fetch and decode units ~~to insert~~ such that micro-opcodes corresponding to interrupt servicing instructions are inserted into an ~~normal~~ instruction sequence within the instruction cache without flushing the instruction cache ~~decoded micro-opcodes representing interrupt servicing instructions for scheduling and execution by said dispatch-and-execute unit~~.

12. (Currently amended) The ~~apparatus~~ processor of claim 11 ~~in which~~

where the computer supports plural hardware interrupt inputs~~,~~;

where the ~~wherein said~~ interrupt-handl~~inger mechanism~~ includes an interrupt concentrator ~~for determining~~ to determine priority among the plural hardware interrupt inputs and ~~for scheduling~~ to schedule the plural hardware interrupt inputs~~,~~;

where the interrupt-handl~~inger mechanism instructing said fetch-and-decode~~ instructs the fetch and decode units ~~to insert~~ such that plural instances of such decoded micro-opcodes are inserted, each instance representing one or more corresponding sets of interrupt servicing instructions, into the normal instruction sequence for serial scheduling and execution of the plural instances of such decoded micro-opcodes by ~~said dispatch-and-execute~~ the dispatch and execute units in accordance with the determined priority of ~~said~~ the plural hardware interrupt inputs.

13. (Currently amended) The ~~apparatus~~ processor of claim ~~10~~ 11 ~~which further comprises~~ comprising:

a context switch prediction unit coupled with ~~said fetch-and-decode~~ the fetch and decode units to predict ~~predicting~~ a naturally occurring context switch and ~~for signaling said~~ to signal the interrupt-handl~~inger mechanism~~ upon such prediction, said context switch prediction unit to analyze one or more instructions within an instruction cache for context switch-inducing instructions,

said interrupt-handl~~inger~~ mechanism ~~responsive~~ coupled to such signaling from said prediction unit instructing ~~said fetch-and-decode~~ the fetch and decode units.

14. (Currently amended) An apparatus, ~~Apparatus for allocating processor bandwidth to hardware interrupts in a computer~~ comprising:

an instruction queue ~~mechanism staging~~ to stage p-code for execution;

a plurality of interrupt inputs ~~signals for coupling to~~ corresponding to ~~one or more~~ a plurality of hardware input/output devices;

interrupt priority logic ~~for ranking~~ to prioritize ~~the relative priority of~~ a plural interrupt corresponding to the plurality of interrupt inputs ~~represented by the plural interrupt signals~~;

bandwidth allocation logic ~~for determining the~~ to allocate processing resources to service the plurality of interrupt according to their relative priority ~~timing by which a ranked one or more of the plural interrupts is to be serviced~~;

a p-code insertion mechanism ~~for inserting~~ to insert interrupt servicing p-code in ~~accordance with the determined timing~~ into ~~said~~ the instruction queue ~~mechanism~~ according to the processing resources allocated to the plurality of interrupt inputs ~~for processing; and~~

a p-code processor coupled ~~with said~~ to the instruction queue ~~mechanism~~ and ~~responsive to said~~ to the insertion mechanism~~, said p-code processor executing~~ to execute ~~p-code within said instruction queue mechanism, the executed p-code including~~ the interrupt servicing p-code

an ~~in-order~~ retirement unit ~~retiring~~ to retire instructions back to said instruction cache.

15. (Currently amended) The apparatus of claim 14,

where~~in said~~ the instruction queue ~~mechanism~~ includes an instruction cache and an in-order instruction ~~fetch-and-decode~~ unit ~~performing~~ to perform branch predictions; and

where~~in said~~ the p-code processor includes a dispatch and out-of-order execution unit

23. (Currently amended) The article of ~~A computer-readable medium containing a program according to~~ claim 22 ~~wherein the program further comprises~~ comprising:

signaling ~~firmware coupled with said detection firmware for signaling said insertion firmware~~ with a vector representing an interrupt service request upon ~~a determination~~ determining ~~by said detection firmware~~ that an interrupt service request of a priority that exceeds a given threshold has occurred~~,~~; and

~~said insertion firmware~~ fetching interrupt service instructions from memory in accordance with the vector representing the determined priority interrupt service request.

24. (Cancelled)

25. (Currently amended) A computer system, comprising:

one or more input/output (I/O) devices ~~each capable of generating~~ to generate one or more hardware interrupts;

an interrupt concentrator to rank and queue the interrupts into an ordered interrupt array;

a core processor coupled ~~with said~~ to the interrupt concentrator, ~~said~~ the core processor including an instruction cache, a ~~fetch-and-decode~~ fetch unit, and a decode unit, ~~said fetch-and-decode~~ the fetch unit to fetch ~~fetching~~ instructions from ~~said~~ the instruction cache and ~~decoding~~ the decode unit to decode the instructions into micro-opcodes, ~~said~~ the core processor ~~further~~ including a ~~dispatch-and-execute~~ dispatch unit to schedule the micro-opcodes and an execute unit having one or more execution ports~~, said dispatch-and-execute unit scheduling and executing~~ to execute the micro-opcodes in ~~said~~ the one or more execution ports and thereafter ~~retiring~~ retire the micro-opcodes back into the instruction cache; and

an interrupt-handl~~inger~~ ~~mechanism~~ responsive to ~~said~~ the one or more hardware interrupts, ~~said~~ the interrupt-handl~~inger~~ ~~mechanism~~ instructing ~~said fetch-and-decode~~ the fetch and decode units to ~~insert~~ signal insertion into an ~~normal~~ instruction sequence decoded micro-opcodes representing interrupt servicing instructions ~~for scheduling and execution~~ to schedule and execute by ~~said~~ the ~~dispatch-and-execute~~ dispatch and execute units.

26. (Currently amended) The computer system of claim 25 which supports plural hardware interrupt inputs, where~~in said~~ the interrupt-handl~~inger~~ ~~mechanism~~ includes an interrupt concentrator ~~for determining~~ to determine priority among and ~~scheduling~~ to

~~scheduling and executing~~ to schedule and execute p-code in parallel among one or more execution ports.

16. (Cancelled)

17. (Currently amended) The apparatus of claim 15 where~~in said~~ the ~~fetch-and-decode~~ in-order instruction unit includes said allocation logic.

18. (Currently amended) The apparatus of claim 17 where~~in said~~ the allocation logic is ~~responsive~~ coupled to a computer operating system.

19. (Currently amended) The apparatus of claim 14 where~~in said~~ the priority logic is ~~responsive~~ coupled to a computer operating system.

20. (Currently amended) The apparatus of claim 14 where~~in said~~ the allocation logic is ~~responsive~~ coupled to a current-usage model.

21. (Currently amended) The apparatus of claim 14 where~~in said~~ the priority logic is ~~responsive~~ coupled to a current-usage model.

22. (Currently amended) A machine-readable medium comprising ~~An article of manufacture for use with a core processor including an instruction queue, the article~~ comprising a computer-readable medium containing ~~a program~~ instructions, that, when executed by a machine cause the machine to perform a method~~, the program~~ comprising:

~~detection firmware for~~ detecting an interrupt service request;

~~insertion firmware for~~ inserting interrupt servicing instructions responsive to the interrupt service request into ~~the~~ an instruction queue; ~~and~~

~~processor firmware for~~ processing the instructions within the instruction queue including the inserted interrupt servicing instructions; and

signaling after detecting an impending natural context switch represented by a recognized instruction sequence within the instruction queue;

where detecting an impending natural context switch is predicated on an instruction stream within the instruction queue of the machine.

schedule the plural hardware interrupts, the interrupt-handl~~inger mechanism~~ instructing ~~said fetch-and-decode~~ the fetch and decode units to insert plural instances of such decoded micro-opcodes, each instance representing one or more corresponding sets of interrupt servicing instructions, into the normal instruction sequence for serial scheduling and execution of the plural instances of such decoded micro-opcodes by ~~said dispatch-and-execute~~ the dispatch and execute units in accordance with the determined priority of said plural hardware interrupts.

27. (Currently amended) The computer system of claim 25 ~~which further comprises~~ comprising:

a context switch prediction unit coupled with ~~said fetch-and-decode~~ the fetch and decode units to predict ~~predicting~~ a naturally occurring context switch represented by a recognized instruction sequence within the instruction cache and ~~for signaling said~~ to signal the interrupt-handling mechanism upon such prediction~~,~~;

where the ~~said~~ interrupt-handling mechanism ~~responsive~~ is coupled to such signaling from ~~said~~ the prediction unit instructing ~~said fetch-and-decode~~ the fetch and decode units.

28. (New) A method, comprising:

detecting an interrupt service request;

inserting interrupt servicing instructions responsive to the interrupt service request into an instruction queue in such manner that the inserted interrupt servicing instructions intervene mainline program instructions within a queue thereby allocating core processor bandwidth between the inserted interrupt servicing instructions and the mainline program instructions;

processing the instructions within the instruction queue including the mainline program instructions and the inserted interrupt servicing instructions, where core processor bandwidth allocation between the mainline program instructions and the inserted interrupt servicing instructions is achieved by concurrent in-line staging of the same within the instruction queue without first flushing the instruction queue of its contents; and

recycling the executed micro-opcodes for optional re-execution thereof in the one or more out-of-order execution units by reordering and retiring the executed micro-opcodes including those micro-opcodes representing the inserted interrupt servicing instructions to the instruction cache.